SOFTWARE MAINTENANCE: RESEARCH AND PRACTICE, VOL. 8, 295–308 (1996)

Research

Methodologies and Software Maintenance

M. J. TAYLOR
School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, L3 3AF, U.K.
A. T. WOOD-HARPER
Information Systems Research Centre, Salford University, Salford, M5 4WT, U.K.

SUMMARY

There are a multitude of information systems methodologies available to practitioners to aid them in the development of new sytstems. In contrast, however, there are very few information systems methodologies aimed specifically at software maintenance, and of the systems development methodologies few pay much attention to software maintenance. This paper is based on case studies in 31 UK IT departments. The research reported in this paper aimed to uncover how information systems methodologies are applied to software maintenance in current practice. The different approaches to applying methodologies to software maintenance are examined together with the reasons for and against using such approaches. Overall methodologies were found to be applied with less rigour and less consistency in software maintenance than in software development. In addition, government and academic initiatives to improve the software maintenance process such as TickiT and the SEI process maturity model appeared to have made little impact on the organizations studied.

KEY WORDS: software maintenance; methodology

1. INTRODUCTION

This paper is concerned with the ways and means by which information systems methodologies are used for software maintenance in current practice. The aim of the case studies carried out in the research reported in this paper was to examine how such usage actually occurs within organizations.

Many surveys have been conducted over the years to establish the percentage of total IT effort that is applied to software maintenance activities (Parikh, 1982; Harrison, 1987; Leonard, Pardoe and Wade, 1988; Nosek and Palvia, 1990). The general view is that typically, maintenance accounts for between 60 and 80 per cent of the total IT activity within a given organization.

Parikh and Zvegintzov (1983) identified software maintenance as the most expensive aspect of the IS life cycle and that therefore, improvement in maintenance activities are even more important than enhancements of the development process. Henry *et al.* (1994) determined in a case study at the General Electric Corporation that only a five per cent

CCC 1040–550X/96/050295–14

Received 24 March 1996
Revised 1 July 1996

maintenance project-level saving would be worth hundreds of thousands of dollars per maintenance project, and that other organizations could also realize similar savings. Boehm (1987) estimated that by 1995, a 20% improvement in software productivity would be worth $90 billion worldwide. The cost of software maintenance in Western Europe was estimated to be £26.5 billion in 1990 (Goodwin 1991).

Dekleva (1992) identified lack of maintenance methodologies, as being one of the top ten problems in software maintenance. However, the delphi study of software maintenance performed by Dekleva, involved three sequential questionnaire surveys of the same group of 67 software professionals over a period of time. Response rates ranged between 58% and 71%. In these three surveys respondents were asked to state the major problems in software maintenance. No actual investigation of the usage of methodologies for software maintenance was performed, the surveys merely highlighted the fact that this was a problem area. Haziza *et al.* (1992) concluded that formalization of the software maintenance process is generally much poorer than that of the development process, although this particular research was based on only eight case studies, all of which were in the aerospace and nuclear industries. In addition this particular research did not investigate actual methodology usage for software maintenance, but again merely highlighted it as a problem area.

The motivation behind this research was that if software maintenance is so costly and so much of a problem to organizations, why is more effect not expended on attempting to improve the activity of software maintenance, particularly with regard to the use of methodologies? Chapin (1987) further discusses the motivation behind research in the process of software maintenance.

In the systems development environment effectiveness and efficiency can be achieved (or at least aimed for) by means of information systems methodologies. New technologies such as fourth generation languages, object orientation and reuse tools can improve the effectiveness and efficiency of IT activities, but unless their usage is properly co-ordinated and controlled, they may not yield the expected benefits. Information systems methodologies can provide the co-ordination and control to apply such new technologies effectively. Longworth (1985) identified over 300 Information System Development Methodologies. In contrast however there are very few methodologies aimed specifically at software maintenance (IEEE, 1992; Vinje, 1991; Watchel, 1991), and of the systems development methodologies only a small handful pay much attention to software maintenance (SSADM, 1990; Boehm, 1988).

In the software maintenance environment previous research had hinted that there was often a haphazard approach to the use of information systems methodologies to support software maintenance activities, although few actual case studies of methodology usage for software maintenance had been undertaken. Swanson and Beath (1991) did however, conduct a questionnaire-based study of 12 US organizations, one of the many conclusions of which was that methodology usage for maintenance was weak in those organizations studied. However, this study did not analyse the approaches by which methodologies were applied to maintenance, and the reasons behind such approaches. Peveratt (1991) stated that most IT managers recognize the need to develop software maintenance methodologies but find it difficult to cost justify such activities. This research based on case studies in 31 UK IT departments aimed to analyse the way in which information systems methodologies were applied to software maintenance in current IT practice.

2. RESEARCH METHOD

This research was based upon case studies in 31 UK IT departments. The IT departments studied were from both the public and private sectors and covered a range of industrial activities. Table 1 shows the industrial/public sectors of the organizations researched.

The size of the IT departments chosen ranged from six IT staff to over 300 IT staff. The main research technique used was to interview at least three senior members of IT staff in each organization regarding the manner in which they applied or did not apply an information systems methodology to their software maintenance projects. By interviewing at least three IT staff in each organization, a coherent and consistent description of information systems methodology usage for software maintenance within the organization can be obtained, and personal perceptions of those interviewed can be, to some extent, eliminated. In addition, the terminology and context associated with information systems methodology usage can be determined and catered for. It should be noted of course that no two IT practitioners will have exactly the same perception of the information systems methodology used within an organization. Therefore this research aimed to find the 'common denominator' between those interviewed within each organization. Standards manuals and inspections of IT artefacts helped to establish the 'common denominator' sought by the researchers. Detailed notes were made by the researchers in each interview situation and were combined with examples of work and standards manuals where the organization's confidentiality requirements permitted.

Questions applied by the researchers in the case studies are given below. It should be noted of course that these had to be varied to cater for organization specific terminology. For example, methods, techniques, best practice and standards were all terms used to describe methodologies in the organizations researched. This is the kind of problem that more superficial methods of data gathering such as questionnaires may flounder upon.

Questions:

- What methodology is applied to software development?
- How rigorously/consistently is the development methodology followed?
- What methodology is applied to software maintenance?
- How rigorously/consistently is the maintenance methodology followed?

Table 1. Industrial/public sectors of organizations researched

Engineering	2
Manufacturing	8
Computer services	2
Financial services	8
Higher education	1
Distribution	4
Defence	1
Local government	2
Civil service	1
Public utility	1
Retail	1
Total	31

- What are the benefits of using methodologies?
- What problems arise through a lack of usage of methodologies for maintenance?
- Why are methodologies not always used for maintenance?

More superficial methods of research such as questionnaires may have difficulty in assessing IT activities in organizations because they may have difficulty in catering for context and terminology differences between organizations. In addition with a typically low response rate, questionnaires may yield skewed results. This is however, still the research method employed by most commercial survey organizations, since they require cheap information gathering techniques and are content to sacrifice a deeper understanding of the subject domain for commercial reasons.

In addition to gathering a wide base of data from the 31 organizations researched, the researchers also performed in depth case studies in four of the organizations involving several site visits over at least a one year period. Swanson and Beath (1991) argue that a multiple case study approach can yield more generalizable results than a single case study approach.

There is a broad spectrum of research methods available to information systems researchers (Wood-Harper, 1989; Douglas, 1976). However, the case study approach was used for this research because the researchers and others believe that: case studies allow the study to be done in a natural setting, learning about the state-of-the-art, and generating theories for practice; that the nature and complexity of the process can be studied; and that situations which are new and rapidly changing can be studied using the approach (Benbasat, Goldstein and Mead, 1987; Jenkins, 1985).

Gummesson (1991) states that an important advantage with case study research is the opportunity for an holistic view of a process: the detailed observations entailed in the case study method enable us to study many different aspects, examine them in relation to each other, view the process within its total environment and also utilize the researchers capacity for understanding. Consequently case study research provides us with a greater opportunity than other available research methods to obtain an holistic view of a specific research project.

Researchers have criticized the lack of empirical research on information systems in real organizational contexts (Jenkins, Naumann and Wetherbe, 1984). More research is needed into the actual practice of information systems development and maintenance in organizations, justifiable even solely on the basis that practice has often preceded theory in the field. Programming style, compiler writing, user interface design are all areas where practice led theory (Glass, 1991). The Sage missile defence system and the Sabre airline reservation system, developed in the 1950s and 1960s, were both examples of sophisticated interactive systems which far exceeded the maturity of the theory at the time (Shaw, 1990). Researchers have criticized the gap between theory and practice, whereby theorists and practitioners are isolated from each other and move in different directions (Chang, 1990).

3. BENEFITS OF METHODOLOGIES

During the course of this research in the 31 organizations studied, numerous methodologies were encountered (Figure 1). In five of the organizations researched more than one information systems methodology was in use in the IT department. This research was

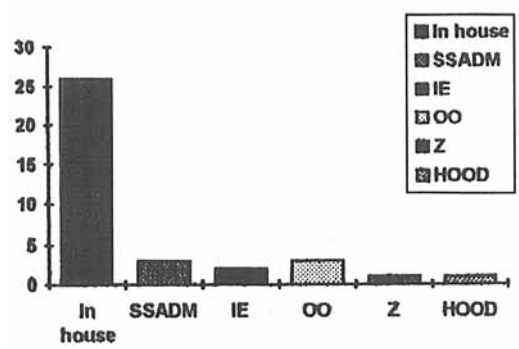

Figure 1. Methodologies used by the organizations studied

concerned with information systems methodologies in general, rather than specific methodologies such as SSADM. In order to understand the context of the usage of methodologies within the organizations studied the researchers investigated the perceived benefits of information systems methodologies usage by the practitioners interviewed. This could then set the background against which the actual usage could be interpreted.

The main perceived benefits of methodology usage encountered in the organizations researched were as follows:

(1) Consistency—this ensures that different teams working on the same project will be able to produce compatible artefacts, and that work can be passed from one team to another for completion. Based upon the interview responses and examination of standards manuals and examples of IT artefacts, in all but four of the 31 organizations studied, the information systems methodology in use was detailed enough to allow for production of compatible IT artefacts. All but three of the organizations studied regarded consistency as a benefit of using methodologies.

(2) Completeness—an information systems methodology should hopefully ensure that all necessary views of the system have been considered and that the design and implementation will contain all the necessary functionality and attributes e.g., the three views of the system: functions, data and the effects of time that are intrinsic to SSADM (1990) philosophy. However, the most important aspect of any information systems methodology should be to satisfy the requirements of the user. One of the shortcomings of SSADM (1990) that was noted by three of the thirty one organizations studied that used SSADM, was that it did not give the users a central role, and hence the identified requirements might be different from what the users actually wanted. Only three of the organizations studied reported any perceived problem with the coverage provided by their methodology. In all three cases this was concerned with the development and maintenance of graphical user interfaces, for which the current methodology provided little support.

(3) Efficiency—if the information systems methodology is correctly chosen and implemented then it should lead to systems of the appropriate quality standard being implemented in the shortest possible time scale. *Ad hoc* methods of development are more likely to miss or confuse requirements as they will not address a given problem in so systematic a manner. Practitioners in 30 of the 31 organizations studied thought that methodologies could aid efficiency.

For further discussion of the perceived benefits of information systems methodologies see Moynihan (1993).

4. THE USAGE OF INFORMATION SYSTEMS METHODOLOGIES FOR SOFTWARE MAINTENANCE ACTIVITIES

Although the practitioners interviewed confirmed the previously mentioned benefits of information systems methodologies, having studied the way in which information systems methodologies were applied to software maintenance activities, this research found that in 23 of the 31 IT departments researched, information systems methodologies were not applied to software maintenance activities in a particularly rigorous or consistent manner, and were certainly applied with less rigour and consistency in software maintenance than software development. A useful metric for assessing the maturity of both development and maintenance processes is the capability maturity model (Paulk *et al.*, 1993). Unfortunately only four of the practitioners out of all those interviewed were even aware of the model, and the model was not actually utilized in any of the 31 organizations studied. However, it still provided at least a benchmark for the researchers.

Rigour of usage of methodologies for IT projects was taken to be that each given technique utilized in an IT project was applied to the standards laid down within the organization. This is more or less equivalent to level 3, the defined level of the capability maturity model (Paulk *et al.*, 1993), whereby there are documented standard software processes. Of course it should be stated that this measure/interpretation of rigour was gained partly from the practitioners interviewed and partly from the standard manuals and examples of IT artefacts inspected in the organizations researched. However, 'pure academic' measurements are difficult to achieve in a fluid working commercial environment, and hence, although bound by organizational context, the interview responses and examples of standards manuals and IT artefacts examined can and do represent a valid assessment of the rigour of methodology usage for software maintenance in those organizations studied.

By consistency it was meant that a given technique would always be used in every IT project or else the same criteria would be used for all projects to decide whether or not the technique should be used. This is the equivalent of level 2, the repeatable level of the capability maturity model, whereby similar IT projects would be carried out in a similar manner. Consistency of usage was measured primarily by the interview responses. Analysis of large samples of project documentation to statistically confirm the practitioners responses regarding consistency was not practical given the large number of practitioners employed amongst the 31 organizations studied (just under 2000) and the large number of software maintenance projects undertaken by these organizations each year (tens of thousands).

When the practitioners were interviewed with regard to the problems that in their experience arose through a lack of usage of the organization's methodology for maintenance the following main points arose:

(1) Time and effort may be required to correct errors introduced by software maintenance activities done without a methodological approach. Roughly 70 per cent of respondents quoted correction of data errors as a problem stemming from a lack of a methodological approach to maintenance projects. As an example, in one finance company researched, incomplete data analysis performed during a software

maintenance project meant that an extra field had to be added to a database at a later date. Owing to the size of the database in question, the subsequent conversion exercise took four hours of mainframe processing to correct the data error.

(2) Even if the software maintenance activity leads to a successful implementation of system amendments, if the system structure is made more difficult to follow due to poor application of design and coding methods then further amendments will be made all the more difficult due to the decrease in understandability of the system. Almost 50 per cent of the respondents indicated that systems had deteriorated through lack of a methodological approach to maintenance. An example of this was encountered at an insurance company where the main control module for one of the company's systems had become so complex after years of *ad hoc* amendments, that it took, on average, three days for an analyst programmer unfamiliar with the module to make even a trivial one line amendment.

(3) Future organizational requirements undermined even though the system works immediately after the amendment is made. Roughly 20 per cent of those interviewed had encountered such problems arising from an incomplete analysis of maintenance requirements. An example of this was encountered by the researchers in a software maintenance project for a stock control system that unknowingly allowed the transfer of stock between warehouses although this was contrary to organizational requirements. The operators of the system, when they found out that they could transfer stock between warehouses, sometimes did so, in order to 'cover up' shortfalls of stock in their particular warehouses. The result was that after a period of time the same physical stock was, according to the system, in two different physical locations, due to a design flaw that had been introduced in the software maintenance project.

(4) Maintenance projects abandoned due to lack of a feasibility study. Just over half of the practitioners interviewed had encountered maintenance projects that were abandoned for a variety of reasons that were not detected at the project outset due to a lack of a feasibility study. An example of this was encountered at a finance organization, where a maintenance project to a reporting suite had to be abandoned, because the fourth generation language used for the reporting suite could not actually support all of the requirements of the project.

Having discovered some of the problems that can occur in software maintenance projects due to the lack of use of information systems methodologies, the research turned to the reasons why information systems methodologies were not used more thoroughly for software maintenance. The main reasons indicated for the lack of use of information systems methodologies for software maintenance projects uncovered by this research included:

(1) Timescales of software maintenance projects. This reason was quoted by around 80 per cent of the practitioners interviewed. In order to respond quickly to user requests, the documentation required for most information systems methodologies was in practice either curtailed or not produced. This is one facet of the 'quick fix' approach (Basili, 1988) to software maintenance. SSADM (1990) in particular was criticized by the practitioners interviewed in the three organizations researched

that used SSADM as being overly bureaucratic, and requiring too muich documentation, some of which was rarely found to be useful anyway.

(2) Costing of software maintenance projects. This reason was indicated by roughly 60 per cent of the practitioners interviewed. The majority of the cost of software maintenance projects is the time of IT staff involved. Most information systems methodologies require a large amount of time to follow the steps in the methodology and produce and maintain the documentation. For example, in five of the organizations researched where there was very strict budgetary control of software maintenance projects, the practitioners indicated that curtailing the use of the information systems methodology was often the only way of staying within budget.

(3) Staff resistance to the use of information systems methodologies. This research found that in five of the 31 organizations studied, IT staff who had been carrying out software maintenance projects informally for many years, displayed a resistance to changing their working practices. Staff resistance to methodologies generally was noted in a study by Bjorn-Anderson *et al.* (1986), and more specifically in software maintenance by Layzell and Macaulay (1990), although the study performed by Layzell and Macaulay was based on only five organizations.

5. APPROACHES TO UTILIZING INFORMATION SYSTEMS METHODOLOGIES FOR SOFTWARE MAINTENANCE ACTIVITIES

Most information systems methodologies are aimed mainly at developing systems rather than maintaining them (Avison and Fitzgerald, 1995). Some information systems methodologies such as Object Oriented Software Engineering (Jacobson *et al.*, 1992) consider future maintenance in the development process but do not specifically address how the methodology should be used in a software maintenance environment. However, a few information systems methodologies do cater for software maintenance with the spiral model of the software life cycle (Boehm, 1988) which makes no distinction between maintenance and development, and SSADM (1990), which prescribes a mandatory subset of the methodology be applied to software maintenance projects, being the most notable Euromethod (1994) which provides a framework for utilizing European government backed information systems methodologies such as SSADM (1990), MERISE and DAFNE could also claim to support software maintenance, but in the 31 organizations studied, few of the practitioners interviewed were even aware of its existence. This is not particularly suprising however, since Euromethod is still in the trial stage, and has only been used in a relatively small number of pilot organizations.

The Tickit guidelines and BS5750/ISO 9000 certification should ensure that for a company having received certification, the software maintenance process would be properly defined. However, these were found to be poorly applied in practice. Of the 31 organizations studied, only ten had bothered to apply either Tickit or BS5750/ISO 9000. In the majority of these ten cases studied, the standards manuals examined were little more than a framework, and contained no actual descriptions of how software maintenance activities should be carried out. The SEI process maturity model (Paulk *et al.*, 1993) was unknown to all but four of the practitioners interviewed, and was not actually applied in any of the 31 organizations studied.

This research based on case studies in 31 UK IT departments suggested that there are five main approaches to utilizing information systems methodologies for software maintenance activities:

(1) Apply the whole information systems methodology for every software maintenance project—this is very thorough, but also extremely time consuming, very costly and typically inappropriate. None of the 31 organizations studied were found to use this approach. The main objections stated to this approach by the practitioners interviewed were that in order to follow all the steps in the methodology, both the elapsed time and the cost would be too high for the majority of software maintenance projects within their organization.
(2) Use the information systems methodology in a contingency fashion—here each technique and method in the information systems methodology would be considered for use in the software maintenance activity, but only those deemed necessary would actually be utilized. This is the approach advocated by the Multiview methodology (Avison and Wood-Harper, 1990). This is less time-consuming than the first approach, and if done properly ensures completeness, but there is the danger that the consideration of the suitability of many of the techniques will be skimped or ignored. This approach to information systems methodology usage for software maintenance projects was adopted in 13 of the 31 organizations studied.

The most rigorous example of this approach was encountered at a financial services company. In this particular company, the appropriate subset of the information systems methodology to be used for a given software maintenance project had to be defined and justified by IT staff, and agreed to by the relevant users before any work on the project could commence. In this particular organization comparisons with similar historical software maintenance projects indicated that a contingency approach to information systems methodology usage could achieve software maintenance project productivity gains of up to ten per cent. However, because precise comparisons could not be made between current and historical projects as no two software maintenance projects were ever exactly the same, this was the IT department's estimate rather than a statistical fact. As a simple example of the contingency approach in action, if a software maintenance project involved amending a report or a screen layout with no change in data requirements, then data analysis techniques within the methodology would be unnecessary for the project.

(3) A subset of the techniques and methods available in the information systems methodology be made compulsory for software maintenance activities. This is the approach recommended in SSADM (1990) for maintenance of systems developed using SSADM. If it is possible to define the 'key' techniques that are necessary to keep the system within acceptable quality levels, then this can prove a cost effective way of ensuring that at least some consistency is applied in the software maintenance environment. However, there is always the danger that some aspect of the system that is not 'modelled' in the 'key' techniques will be overlooked in the software maintenance activity and lead either to system errors or system degradation. Four of the 31 companies studied used this approach for their software maintenance projects. As only one of the four IT departments using the compulsory subset approach to information systems methodology usage for software maintenance

projects had historical data for software maintenance projects performed not using this approach, a truly objective appraisal of its effectiveness was unfortunately not possible. However, all the IT staff interviewed in these four organizations stated that they considered it to be a productive approach and in the one organization which did have historical data through which comparisons could be made, there was a definite reduction in software maintenance project timescales for similar projects.

(4) Have a separate software maintenance methodology (IEEE, 1992; Vinje, 1991; Watchel, 1991). This approach can have problems because the maintenance and development methodologies may not be completely compatible, and either staff will have to be trained in both methodologies incurring time and cost, or else staff will be trained either for software development or software maintenance in which case flexible working patterns and career development opportunities wil be lost. Seven of the 31 organizations studied had a separate software maintenance methodology. However, in the majority of these seven organizations, the maintenance methodology defined in the organization's standards manuals was more concerned with the management and control of software maintenance projects, rather than the techniques to be used for analysis of maintenance requirements and design of changes to the relevant systems. For example, in one manufacturing company researched, the software maintenance methodology, as such, consisted of a maintenance request specification, a test plan and test case procedure, and an authorization to go live procedure. Somewhat surprisingly,although this IT department had such a meagre software maintenance methodology, they were fully BS5750 accredited for manufacturing.

(5) Have a software maintenance methodology which has its own specific techniques, but which also utilizes some of the techniques of the software development methodology. Five of the 31 IT departments studied applied this approach in practice. As an example, one of the organizations researched employed a formal systems acceptance methodology (Ball, 1985) to ease the transition from systems development to systems maintenance. In this organization, dataflow diagrams, entity models and pseudo-code were used for development projects, but only pseudo-code was used for maintenance projects. The reason for this was stated as being that the overhead of amending diagrams (no CASE tool was in use) was too great, although such diagrammatic techniques were beneficial when developing systems in order to confirm user requirements.

Figure 2 shows the relationships between methods and techniques for software development and software maintenance in current practice uncovered by this research. Case (a) in Figure 2 indicates the instance of specific techniques existing for software maintenance, but the use of some software development techniques also occurring for software maintenance. Case (b) shows the instance of software maintenance methods being a subset of the software development methods with the subset being used in either a mandatory fashion or a contingency fashion. Case (c) indicates separate methods for software development and software maintenance. Case (d) shows the same set of techniques and methods being used for both software development and software maintenance.

As regards the overall effectiveness of using the different methodological approaches to software maintenance in practice, the researchers found that only seven of the organizations studied kept any form of statistics on their software maintenance projects. Typically

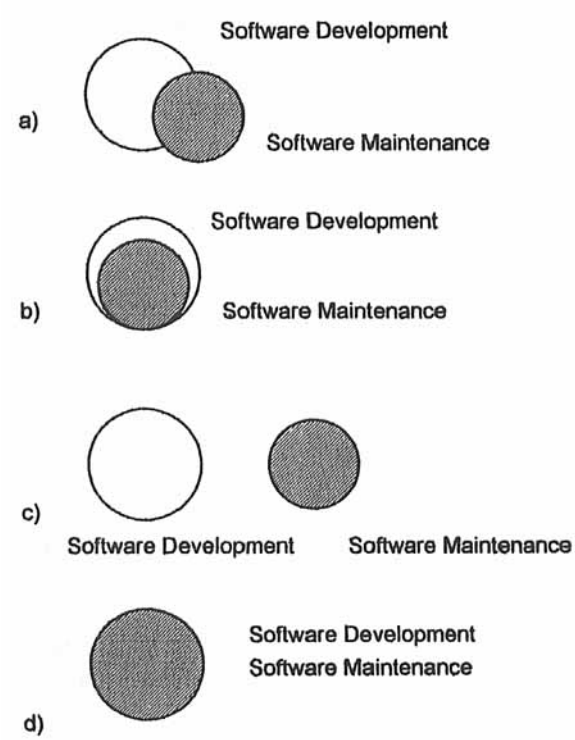

Figure 2. Relationship between methods and techniques for software development and software maintenance in practice

this was purely limited to the estimated number of person hours and the actual number expended for a given software maintenance project. As the majority of these seven organizations had been using the same methodological approach to software maintenance projects for a number of years, truly objective comparisons to software maintenance projects performed not using the same methodological approach were not possible. However, in one of the organizations studied, which had been studied over a two year period, and which had introduced a new organization-wide methodology in the time frame studied, a definite reduction in the time for completion of software maintenance projects was quoted by the practitioners interviewed, as a result of applying a mandatory subset of their development methodology to software maintenance projects.

Another in depth case was investigated at a large financial services organization that was studied on several occasions over a 14 month period. This particular organization used a formalized contingency approach, whereby the subset of the information systems methodology to be used for software maintenance projects had to be defined, justified and agreed before the project could commence. In this organization, software maintenance projects were compared with previous similar projects to aid with timescale and cost estimates, and to identify suitable ways in which to tailor the information systems methodology for the project in hand. In this organization there was a continual drive to improve the productivity of IT staff by harnessing the most appropriate aspects of the information systems methodology for a given software maintenance project. The IT management's belief that new software maintenance projects could be completed in shorter

timescales and in smaller budgets than previous similar projects paved the way for appropriate tailoring of the methodology for software maintenance projects to yield productivity gains, and estimates in this organization showed a roughly ten per cent productivity increase.

6. CONCLUSIONS

In this paper we have outlined the state of information systems methodologies usage for software maintenance as found in current practice by case studies in 31 UK IT departments. The reasons why information systems methodologies were applied less thoroughly to software maintenance than software development have been discussed along with the resulting problems that may occur. This research has identified the main approaches to utilizing information systems methodologies within the software maintenance environment, found in current practice, along with some of the advantages and disadvantages of each approach. Further research is required in this area in order to attempt to reduce the high cost of software maintenance activities, and at least bring productivity in software maintenance to the level of productivity in software development (Arthur, 1988). Research seminars involving representatives from both industry and academia are one mechanism for spreading 'best practice' amongst companies, and for transferring new theories from academia to industry. The lack of use of TickiT/BS5750 and the SEI process maturity model highlighted the fact that organizations were slow to respond to existing government and academic initiatives to improve the software maintenance process.

References

Arthur, L. J. (1988) *Software Evolution*, John Wiley, New York, U.S.A.

Avison, D. E. and Fitzgerald, G. (1995) *Information Systems Development: Methodologies, Techniques and Tools*, McGraw Hill, Maidenhead, U.K.

Avison, D. E. and Wood-Harper, A. T. (1990) *Multiview: An Exploration in Information Systems Development*, Blackwell Scientific Publications, Oxford, U.K.

Ball, R. K. (1985) 'Managing the transition from development to support', *Data Management*, March 1985.

Basili, V. R. (1988) 'Maintenance = reuse-oriented software development', in *Proceedings of Conference on Software Maintenance*, Scottsdale, AZ, 24–27 October 1988, IEEE Computer Society Press, Washington DC, U.S.A., pp. 160–169.

Benbasat, I., Goldstein, D. and Mead, M. (1987) 'The case research strategy in studies of information systems', *MIS Quarterly*, February 1987.

Bjorn-Andersen, N., Eason, K. and Robey, D. (1986) *Managing Computer Impact: An International Study of Management and Organisations*, Ablex, New Jersey, U.S.A.

Boehm, B. (1987) 'Improving software productivity', *IEEE Computer*, **9**, 43–57.

Boehm, B. (1988) 'A spiral model of software development and enhancement', *IEEE Computer*, **21**(5), 61–72.

Chang, C. (1990) 'Editor's message: let's stop the bipolar drift', *IEEE Software*, **7**(3), 4.

Chapin, N. (1987) 'The job of software maintenance', in *Proceedings of Conference on Software Maintenance*, Austin, TX, U.S.A., September 21–24, 1987, IEEE Computer Society Press, Washington, DC, pp. 4–12.

Dekleva, S. M. (1992) 'Delphi study of software maintenance problems', in *Proceedings of Conference on Software Maintenance*, 9–12 November, 1992, Orlando, Florida, IEEE Computer Society Press, Los Alamitos, CA, U.S.A., pp. 10–17.

Douglas, J. D. (1976) *Investigative Social Research*, Sage, Beverly Hills, California, U.S.A.

Euromethod (1994) Euromethod Information Office, Excelsiorlaan 48/50, 1930 Zaventem, Belgium.

Glass, R. (1991) *Software Conflict: Essays on the Art and Science of Software Engineering*, Yourdon Press, Prentice Hall, Englewood Cliffs, New Jersey.
Goodwin, C (1991) 'Reputation in for repair', *Computing*, 15 August 1991.
Gummesson, E. (1991) *Qualitative Methods in Management Research*, Sage, Beverley Hills, CA, U.S.A.
Harrison, R. (1987) 'Maintenance giant sleeps undisturbed in federal data centres', *Computerworld*, 81–86.
Haziza, M., Voidrot, J., Minor, E., Pofelski, L. and Blazy, S. (1992) 'Software maintenance: an analysis of industrial needs and constraints', in *Proceedings of Conference on Software Maintenance*, 9–12 November, 1992, Orlando, Florida, IEEE Computer Society Press, Los Alamitos, CA, U.S.A., pp. 18–26.
Henry, J., Henry, S., Kafura, D. and Matheson L. (1994) 'Improving software maintenance at Martin Marietta', *IEEE Software*, **11**(4), 67–75.
IEEE (1992) 'Draft standard for software maintenance (P1219), *IEEE*, New York, U.S.A.
Jacobson, I., Christerson, M., Jonsson, P. and Overgaard, G. (1992) *Object-Oriented Software Engineering—A Use Case Driven Approach*, Addison Wesley, Reading, MA, U.S.A.
Jenkins, A. M. (1985) 'Research methodologies and MIS research', in Mumford, E. *et al.* (Eds) *Research Methods in Information Systems*, North Holland, Amsterdam, The Netherlands.
Jenkins, A., Naumann, J. and Wetherbe, J. (1984) 'Empirical investigation of systems development practices and results', *Information and Management*, **7**, 73–83.
Layzell, P. and Macaulay, L. (1990) 'An investigation into software maintenance perception and practices', in *Proceedings of the Conference on Software Maintenance*, 26–29 November, 1990, San Diego, California, IEEE Computer Society Press, Los Alamitos, CA, U.S.A., pp. 130–140.
Leonard, S., Pardo, J. and Wade, S. (1988) 'Software maintenance—Cinderella is still not getting to the ball, in *BCS/IEE Conference on Software Engineering*, Liverpool Polytechnic, Conference Publication 290, University of Liverpool, Liverpool, U.K., pp. 104–106.
Longworth, G. (1985) *Designing Systems for Change*, NCC, Manchester, U.K.
Moynihan, E. (1993) *Business Management & Systems Analysis*, Alfred Waller, Henley on Thames, U.K.
Nosek, J. and Palvia, P. (1990) 'Software maintenance management: changes in the last decade', *Journal of Software Maintenance: Research and Practice*, **2**, 157–174.
Parikh, G. (1982) *Techniques of Program and System Maintenance*, Winthrop Publishers, Cambridge, MA, U.S.A.
Parikh, G. and Zvegintzov, N. (1983) *Tutorial on Software Maintenance*, IEEE Computer Society Press, Silver Spring, MD, U.S.A.
Paulk, M. C., Curtis, B., Chrississ, M. B. and Weber, C. V. (1993) 'Capability maturity model version 1.1', *IEEE Software*, **10**(4), 18–27.
Peveratt, T. (1991) 'Managing the maintenance crisis', *Infomatics*.
Shaw, M. (1990) 'Prospects for an engineering discipline of software', *IEEE Software*, 15–24.
SSADM (1990) SSADM Developers Handbook, version 4, NCC Blackwell, Oxford, U.K.
Swanson, E. and Beath, C. (1991) *Maintaining Information Systems in Organisations*, Wiley, Chichester, U.K.
Vinje, P. (1991) *Aktiv Systemforvaltning*, Teknisk Forlag, Copenhagen, Denmark.
Watchel, R. (1991) *The Software Modification Manual*, North Bay Software Analysis, Occidental, California, U.S.A.
Wood-Harper, A. T. (1989) 'Comparison of information systems definition methodologies: action research multiview perspective', Ph.D. Thesis, University of East Anglia, Norwich, U.K.

Authors' biographies:

Mark Taylor is currently a Senior Lecturer at Liverpool John Moores University. In the course of his industrial career he has worked in both the manufacturing and finance sectors as an Analyst Programmer, Systems Designer, and Systems Analyst, and is currently an active IT consultant. He is a member of the British Computer Society.

Trevor Wood-Harper is currently Professor of Computer Science at Salford University and heads the research group in information systems methodology and evaluation. He is an active consultant using the Multiview methodology, and has published widely with more than fifty research articles and four books to his credit.